« Previous Next »

Advanced

- UI Extras
- Disk Cache
- Web UI
- Run Program

WARNING: Unless you fully understand the consequences of changing an advanced setting, you should not modify it without guidance.

Notes:

3.3: This option is applicable only from µTorrent version 3.3 and above

TBD: This option still needs to be explained further

Unused: This option shows in the advanced preferences, but is currently not being used

- **bt.allow_same_ip**: Enabling this option allows multiple incoming connections from the same IP address. This option affects a single torrent job at a time, so you can still have the same IP address connect to you on different torrent swarms. It is recommended that this option be left disabled, as it weakens the anti-leech protection.
- **bt.auto_dl_enabled**: TBD
- **bt.auto_dl_factor**: TBD
- **bt.auto_dl_interval**: TBD
- **bt.auto_dl_qos_min**: TBD
- **bt.auto_dl_sample_average**: TBD
- **bt.auto_dl_sample_window**: 3.3
- **bt.ban_ratio**: The lowest acceptable ratio of good to bad pieces a peer can send before it gets banned. The lower this option is set, the more forgiving µTorrent will be toward bad pieces, meaning that it will be less likely to ban a peer. This takes effect after bt.ban_threshold is exceeded and bt.use_ban_ratio is enabled.
- **bt.ban_threshold**: This option specifies the maximum number of hashfailed pieces any single peer can send before µTorrent takes action against it (either banning it outright, or enforcing bt.ban_ratio if bt.use_ban_ratio is enabled).
- **bt.compact_allocation**: Enabling this option allows µTorrent to create files in a manner such that the data are incrementally written to disk without file pre-allocation. Because writes are compact, enabling this option may potentially lead to an increased level of disk fragmentation while the file remains incomplete. In addition, this option further decreases the already-low probability that a file can be previewed before completion, since it may write the data for in-progress files out of order. Here are some things to take note of when using this option:
  - If you tell µTorrent to pre-allocate all disk space, this option is ignored, and µTorrent will pre-allocate the file anyway.
  - If this option is enabled, files can't be skipped. If a torrent job has skipped files, it will not use compact allocation.
- **bt.connect_speed**: This option specifies the number of connections µTorrent should allow to be attempted and/or established each second, whether the connections use uTP or TCP.
- **bt.determine_encoded_rate_for_streamables**: Enabling this option shows media files' rate in Files tab
- **bt.enable_pulse**: Enabling this option effects ratings, and disables comments too.
- **bt.failover_peer_speed_threshold**: TBD 3.3
- **bt.save_resume_rate**: This option set the time interval for saving the "resume" data to every n seconds.
- **bt.enable_tracker**: Enabling this option enables the rudimentary tracker embedded in µTorrent. If you wish to use this tracker, the URL is located at `http://IP:port/announce`, where `IP` is your WAN IP address, and `port` is the port µTorrent is listening on (or the alternative listening port if set and enabled). If you use a dynamic DNS service, your domain may be used instead of your IP address. The embedded tracker allows tracking of external .torrent files, and provides no way to limit them. There is no interface for viewing the .torrent files that are tracked. It is imperative that µTorrent is able to listen for incoming connections for this feature to work properly, so you have to make sure you have completely forwarded your ports in order to use the embedded tracker.
- **bt.graceful_shutdown**: If enabled, µTorrent will take as long as it needs to finish its shutdown sequence (writing in-progress pieces to disk, deleting files in deletion queue, and waiting for tracker replies to stop messages -- among other things). That means that even if it takes several minutes to shutdown gracefully, it will wait for that long, and the process will remain in memory until then. If disabled, µTorrent will limit how long it waits to to `10` seconds, and regardless of the state of the shutdown sequence, µTorrent will force itself to exit.
- **bt.multiscrape**: Enabling this option allows µTorrent to send multiple hashes each time it scrapes a tracker, which is more efficient than sending one hash at a time. In most circumstances, this option should not need to be disabled, as µTorrent will fall back to single scraping if it detects that the tracker does not support multi-scraping.
- **bt.no_connect_to_services**: This option tells µTorrent not to connect to peers using ports specified in bt.no_connect_to_services_list as their listening ports. This stops firewalls from complaining about µTorrent trying to send an e-

mail.

- **bt.no_connect_to_services_list**: This option specifies which ports µTorrent should not connect to when bt.no_connect_to_services is enabled.
- **bt.prio_first_last_piece**: Enabling this option prioritizes the first and last pieces of each file in a torrent job, increasing the chances that they can be previewed before download completion. µTorrent will prioritize at least the first and last 1 MiB of data in a file.
- **bt.prioritize_partial_pieces**: Enabling this option ('True'), tells µTorrent to always try to request blocks from pieces we've already started.
- **bt.promiscuous_mode**: TBD
- **bt.pulse_interval**: Unused 3.3
- **bt.pulse_weight**: Unused 3.3
- **bt.randomize_peer_id**: Disabling this option tells µTorrent not to change peerid on every connection. It still uses unique one for tracker, and should also be random limit. This randomization is not being used for private torrents.
- **bt.ratelimit_tcp_only**: Enabling this option tells µTorrent to limit the upload and download rates for TCP connections based on information received over the uTP transport rather than using static global rate limits. This option is ignored if bt.tcp_rate_control is disabled.
- **bt.rcv_max_sockbuf**: This debugging option defines µTorrent's local receive buffer socket size. Tweaking can reduce memory usage, but may hurt performance if too low. 3.3
- **bt.read_only_on_complete**: This option makes downloaded files have the read-only attribute set when torrent is complete. This prevents MP3s and the like from being retagged by media players and corrupted.3.3
- **bt.save_resume_rate**: This option sets µTorrent to save resume data every n seconds.
- **bt.scrape_stopped**: Enabling this option allows µTorrent to get seed and peer counts for torrent jobs that are stopped.
- **bt.send_have_to_seed**: Enabling this option tells µTorrent to send a message to other seeds indicating how many pieces you currently have.
- **bt.send_max_sockbuf**: This debugging option defines µTorrent's local send buffer socket size. Tweaking can reduce memory usage, but may hurt performance if too low. 3.3
- **bt.set_sockbuf**: This debugging option allows µTorrent to automatically detect the TCP buffer size periodically (so_sndbuf) and adjust it based on your upload speed. It does not adjust based on latency.
- **bt.shutdown_tracker_timeout**: This option controls the maximum amount of time µTorrent will wait, when exiting, for each tracker to respond to a stopped event before it forces itself to terminate. This value is interpreted in seconds, so please enter it as such. Setting this value to 0 tells µTorrent to wait for an indefinite amount of time until it receives a response.
- **bt.shutdown_upnp_timeout**: This option controls the maximum amount of time µTorrent will wait, when exiting, for routers to respond to a request to un-map the listening ports before it forces itself to terminate. This value is interpreted in seconds, so please enter it as such. Setting this value to 0 tells µTorrent to wait for an indefinite amount of time until it receives a response.
- **bt.tcp_rate_control**: Enabling this option tells µTorrent to use information from the uTP transport as hints for limiting TCP transfer rates.
- **bt.transp_disposition**: This option controls µTorrent's level of bias towards using TCP or uTP for transporting data (assuming the peer at the other end of the connection supports both transport protocols). The following is a list of the accepted values:
  - **1** allows µTorrent to attempt outgoing TCP connections
  - **2** allows µTorrent to attempt outgoing uTP connections
  - **4** allows µTorrent to accept incoming TCP connections
  - **8** allows µTorrent to accept incoming uTP connections
  - **16** tells µTorrent to use the new uTP header. This is an improved communication header, but is not backwards compatible with clients that do not understand it.

  This option is interpreted as a bitfield, so values can be added together to obtain a combination of behaviors. Setting this value to 255 guarantees that all behaviors are enabled.
- **bt.use_ban_ratio**: This option tells µTorrent to use bt.ban_ratio to decide when a peer gets banned after it has exceeded bt.ban_threshold.
- **bt.use_dns_tracker_prefs**: This option tells µTorrent to respect or ignore BEP34.
- **bt.use_rangeblock**: When enabled, µTorrent will automatically attempt to determine whether an entire range of IP addresses should be banned for sending hashfailed pieces rather than banning individual IPs one at a time. When µTorrent bans 4 IPs from the same /24 CIDR block, it will ban the entire /24 CIDR block. When µTorrent bans 4 CIDR blocks of size /24 from the same /16 CIDR block, it will ban the entire /16 CIDR block. When µTorrent bans 4 CIDR blocks of size /16 from the same /8 CIDR block, it will ban the entire /8 CIDR block.
- **btapps.app_store**: TBD
- **btapps.apps_chammel**: TBD
- **btapps.apps_update_btapps**: TBD
- **btapps.apps_update_btinstalls**: TBD

- **btapps.enable_activx**: TBD
- **content_offer_autoexec**: When enabled, if torrent offer (featured content) has autoexec flag AND is signed correctly by us, µTorrent will auto-execute the content.
- **content_offer_url**: TBD
- **dht.collect_feed**: When enabled, RSS feeds via DHT. Unused
- **dht.collect_rate**: This option will determine the rate at which RSS feeds via DHT - are being scanned. TBD.
- **dht.rate**: This option specifies the amount of bandwidth that DHT will use. The default value, -1, tells µTorrent to manage the bandwidth usage automatically based on your maximum upload rate. The automatic value is obtained by dividing your maximum upload rate by 16. This value is interpreted in bytes per second, so please enter it as such.
- **diskio.all_writes_sync**: Enabling this option causes µTorrent to open files in synchronous mode so all writes are immediately flushed to disk 3.3
- **diskio.cache_reduce_minutes**: This option determine how often (in minutes) µTorrent compacts the disk cache
- **diskio.cache_stripe**: This option tells µTorrent the size of blocks of memory used in it's disk cache in KiB. Minimum of piece size and cache stripe are in KiB.
- **diskio.coalesce_write_size**: This option determines the size threshold for which µTorrent should write data out coalesced, and is relevant only if diskio.coalesce_writes is enabled. This value is interpreted in bytes per second, so please enter it as such.
- **diskio.coalesce_writes**: This option tells µTorrent to try to minimize the number of writes to disk by writing more data at once. It doesn't have any effect on download speeds, but might increase memory and CPU usage to achieve less disk writes.
- **diskio.flush_files**: Enabling this option causes µTorrent to close file handles every minute. It helps to reduce the effect of Windows managing the system cache badly for some people and causing apparent "memory leaks."
- **diskio.mark_of_the_web**: This option sets NTFS ADS that tells Windows that this file was downloaded from the Internet 3.3
- **diskio.max_write_queue**: This option sets the maximum depth of the write queue before the client starts showing disk overloaded 3.3
- **diskio.minimize_kernel_caching** : This option disables compact allocation, might be POSIX only 3.3
- **diskio.no_zero**: Enabling this option causes µTorrent to skip the zero-filling process for file allocation. This option works only on Windows XP or newer, and requires administrator privileges by default. However, it is possible to make this work on limited accounts by setting the "Perform volume maintenance tasks" policy appropriately in the Windows Group Policy Editor. Skipping zero-filling speeds up the file allocation process, but because the allocated files have shared read access, there is a risk that any sensitive data that may have once existed at that location in disk but isn't wiped will potentially be exposed for other applications and users to read, including those without volume maintenance privileges.
- **diskio.quick_hash** : This option disables compact allocation, might be POSIX only. Unused
- **diskio.resume_min**: TBD
- **diskio.smart_hash**: This option makes µTorrent hash data from memory (if in the write queue) instead of flushing to disk, re-reading from disk, and then hashing. This should help reduce hard disk reads, especially when transferring at high speeds.
- **diskio.smart_sparse_hash**: This option is a workaround for a problem in some versions of Windows that return incorrect data to µTorrent regarding sparse files and the amount of data that has actually been completed on disk.
- **diskio.sparse_files**: Enabling this option causes µTorrent to allocate only the data that it writes, but will inform the filesystem of the file's size (so that it can attempt to reserve enough contiguous space on the hard drive without having to physically zero all of the space out for the file). Even though space is reserved for the file, no space will be taken for the unwritten parts of the file. Enabling this option may potentially lead to increased disk fragmentation in rare cases where the drive does not have enough free space available to honor the space reservation for sparse files. Here are some things to take note of when using this option:
  - Sparse files work only on partitions that are formatted as NTFS.
  - Hash checking sparse files tends to be quicker than hash checking pre-allocated files, as µTorrent won't have to hash zeroed-out pre-allocated data.
  - On Windows Vista, sparse files can cause µTorrent to run into a file system limitation.
  - If you are using a non-administrator account with a disk quota, sparse files won't work, and the file will still get fully allocated. This is a limitation with Windows that µTorrent can't do anything about.
  - This option cannot be used in conjunction with pre-allocate all files. If both options are enabled simultaneously, pre-allocation will take precedence.
  - When used in conjunction with bt.compact_allocation, µTorrent will reserve space for each file in the filesystem, but it will continue to use compact writes.
- **diskio.use_partfile**: This option is used to store data that is downloaded from files that you told µTorrent to skip. This is necessary to prevent the file from being allocated. It separately stores the parts of the skipped files that come with a piece, since µTorrent must download and save the entire piece in order to confirm that it is uncorrupted, and each piece can contain data from multiple files. The partfile is removed when you remove the torrent job from the torrent job list.
- **distributed_share.enable**: This option enables the participation in distributed backups 3.3
- **gui.auto_restart**: This option configures crash recovery in µTorrent. When this option is enabled, if µTorrent crashes while it is minimized and the user has been idle at the computer for more than 1 minute, this option will cause µTorrent to automatically restart, and a notification of the crash (without a crash dump or any personally identifiable information) will be sent to the developers. If µTorrent crashes more than once within an hour, this option will not cause it to automatically restart again after

the first crash, as such frequency of crashes is indicative of some important underlying problem that should not be ignored by the user. In this situation, µTorrent will fall back to the regular behavior (as if this option were disabled), where it displays a crash dialog that allows the user to choose how to proceed.

- **gui.color_progress_bars**: TBD Currently broken...
- **gui.combine_listview_status_done**: This option will show you the progress of the download (% done downloading) inside the colored status bar/column.
- **gui.compat_diropen**: If you experience abnormal behaviors while browsing directories in µTorrent, such as a blank browsing dialog, try enabling this option.
- **gui.default_del_action**: This option tells µTorrent how it should remove torrent jobs when pressing the Remove button or Delete on your keyboard. Note that any value above 3 will cause the "*Remove*" button and the Delete button on your keyboard to do nothing in µTorrent. To be safer, you'd best be setting this option in the GUI through the toolbar method.
  - **0** means "*Remove*"
  - **1** means "*Remove and delete .torrent*"
  - **2** means "*Remove and delete Data*"
  - **3** means "*Remove and delete .torrent + Data*"
- **gui.delete_to_trash**: Enabling this option tells µTorrent to attempt to delete files to the Recycle Bin rather than directly erasing them from the disk. It is easier to set this option in the GUI through the toolbar method.
- **gui.enable_comments**: This option disables the torrent comment feature. You will not be able to edit in connects in the Rateings tab not in the Delete torrent dialogue.
- **gui.enable_ratings**: Unused
- **gui.graph_legend**: This option tells µTorrent to draw a legend over the graphs displayed in the Speed tab to describe each of the lines drawn on the graph.
- **gui.graph_overhead**: If enabled, this option tells µTorrent to draw communication overhead lines in the Speed tab's transfer rate graphs. Otherwise, only the "*Network Overhead*" graph will display information about communication overhead.
- **gui.graph_tcp_rate_control**: This option tells µTorrent to draw the TCP rate control/limit graph on the download speed graphs
- **gui.graphic_progress**: This option tells µTorrent to draw a progress bar for each torrent job in the torrent jobs list, behind the Done column.
- **gui.log_date**: This option causes the date to be included in the timestamp shown in the Logger tab.
- **gui.overhead_in_statusbar**: This optio shows the protocol-network overhead in the status bar.
- **gui.piecebar_progress**: If enabled, this option tells µTorrent to draw the lower Downloaded bar as the progress bar for each torrent job in the torrent jobs list, behind the Done column. This option works only if gui.graphic_progress is enabled, and will hide the percentage from the column.
- **gui.report_problems**: If enabled, this option tells µTorrent to report hangs in the user interface thread back to the µTorrent servers anonymously. The information sent is not personally identifiable, but can assist the developers in fixing (or identifying the cause of) the user interface hang. Whenever a report is sent to the server, a message is added to the Logger tab.
- **gui.show_av_icon**: This option shows an antivirus icon when Plus version is installed.
- **gui.show_devices**: This option show devices pane in sidebar.
- **gui.show_notorrents_node**: This option will show a page in the sidebar when no torrents are loaded.
- **gui.show_player_node**: This option puts a new item in the sidebar when streaming a video in the client.
- **gui.show_plus_upsell**: This option shows the "Plus" box on the sidebar.
- **gui.show_rss_favicons**: This option shows favicons for your RSS feeds
- **gui.show_status_icon_on_dl_list**: This option shows status icons in the main listview of torrents.
- **gui.show_welcome_node**: Unused
- **gui.toolbar_labels**: TBD. Currently broken ...
- **gui.tall_category_list**: This option toggles the Category List's height between short and tall. When taller, the Category List displaces the Detailed Info Pane's left-hand side. When shorter, the Category List's lower section is displaced by the Detailed Info pane. A taller list might be more optimal for users with many labels and RSS feeds
- **gui.transparent_graph_legend**: If enabled, this option tells µTorrent to draw a transparent background behind the legend (otherwise, the background is opaque).
- **gui.update_rate**: This option controls the amount of time between each update of the µTorrent main window. The higher it is, the less frequently µTorrent updates the main window, meaning that if you select 1000, the information displayed on the main window is at most 1000 milliseconds (1 second) old. For users of slower computers, you might want to increase this number to decrease resource usage when the main window is displayed. Any value below 500 will be ignored (and 500 will be used instead).
- **gui.use_fuzzy_dates**: This option shows inexact dates instead of timestamps (e.g. "5 minutes ago")
- **ipfilter.enable**: This option, when enabled, tells µTorrent to load ipfilter.dat and apply the rules on connections established after it is loaded. Note that disabling and re-enabling this option will force µTorrent to reload ipfilter.dat.

- **isp.bep22**: This option enables Local Tracker Discovery accordonmg to BEP22, allowing µTorrent to attempt to discover ISP-local trackers via a series of reverse DNS lookups. The ISP-local tracker can return a list of peers and caches (most likely ISP-local). Note that if your ISP is known to interfere with BitTorrent traffic, careful consideration should be taken in deciding to enable this option. Announcing to a ISP-hosted tracker indicates to the ISP that you are using BitTorrent, and as such, can make it easier for the ISP to interfere. Private torrent jobs are not announced to local trackers.
- **isp.fqdn**: TBD
- **isp.peer_policy_enable**: This option enables peer policy functionality, which sets weights to different IP ranges.
- **isp.peer_policy_override**: This option overrides the peer policy.
- **isp.peer_policy_url**: This option sets a URL to the ISP's peer policy.
- **isp.primary_dns**: This option sets the primary DNS server Ip of your ISP.
- **isp.secondary_dns**: This option sets the primary DNS server Ip of your ISP.
- **left_rail_offer_enabled**: This options enables new "Offers" (Ads) by Bittorrent Inc located at the left pane. Disable it if you like not to be alerted to new offers.
- **logger.log_upnp_to_file**: This option logs debug output of UPnP to a file (warning: spammy).
- **net.bind_ip**: If your computer setup requires that you use a specific LAN adapter for incoming connections, you may specify that adapter's IP address here.
- **net.calc_rss_overhead**: This option applies ratelimits to RSS traffic as well.
- **net.calc_tracker_overhead**: This option applies ratelimits to tracker traffic (warning: could break tracker communication under load).
- **net.discoverable**: If enabled, this option tells µTorrent to listen on one of a sequence of well-known ports for incoming connections in addition to the standard and alternative listening ports. Because the sequence of ports is well-known to applications attempting to interface with µTorrent, it allows for such applications to connect to µTorrent with less effort on the user's part.
- **net.disable_incoming_ipv6**: This option blocks all incoming IPV6 connections.
- **net.freindly_name**: TBD
- **net.limit_excludeslocal**: This option decides whether µTorrent should apply the Transfer Cap limits to traffic between itself and peers on the local network. Peers are considered local if they are discovered by Local Peer Discovery, or if they are on the same LAN as the client.
- **net.low_cpu**: Enabling this option reduces CPU usage slightly. You may achieve faster speeds with this option disabled. In general, this option is useless for most people unless they have extremely fast connections.
- **net.max_halfopen**: This option specifies how many connections µTorrent should attempt to establish simultaneously at any given time. On systems running Windows XP with Service Pack 2 (SP2) or newer, if your TCPIP.sys file is unpatched, you should leave this option at its default value.
- **net.outgoing_ip**: If your computer setup requires that you use a specific LAN adapter for outgoing connections, you may specify that adapter's IP address here. Note that Windows will sometimes ignore this setting and use other adapters due to their binding orders in Windows. To fix this, read Microsoft's knowledge base article KB894564.
- **net.outgoing_max_port**: This sets the upper limit for the outgoing port range. If this option is set to some invalid port number or some value less than net.outgoing_port, it gets ignored, and only net.outgoing_port gets looked at (meaning the outgoing port "range" will actually be a single outgoing port).
- **net.outgoing_port**: This option specifies the port that µTorrent should use to make outgoing connections. Normally, µTorrent selects a port from the ephemeral port range at random. "This can be used with full cone NAT routers to reduce the number of NAT table entries and thus prevent cashes on some router models. When the outgoing port is bound to the same as the incoming port that might even solve NAT problems on full cone NAT routers" (Advanced Network Settings on AzureusWiki). This option only works on Windows 2000 and above. This option is ignored if it is not a valid port number.
- **net.upnp_tcp_only**: This option disables automatic forwarding of the listening port for UDP via UPnP, telling µTorrent to forward the port for TCP only. This fixes an issue with some broken routers that overwrite the TCP forwarding with the UDP forwarding.
- **net.utp_dynamic_packet_size**: If enabled, this option allows µTorrent to adjust the uTP packet size in response to connection conditions detected through information gathered by uTP, changing up to as often as net.utp_packet_size_interval allows. If disabled, µTorrent uses the initial packet size for all uTP communication, as set by net.utp_initial_packet_size.
- **net.utp_initial_packet_size**: This controls the initial size of the uTP packets that µTorrent uses when initiating a uTP connection. If net.utp_dynamic_packet_size is enabled, packet sizes can change dynamically during the lifetime of the uTP connection, depending on the connection conditions; this option only controls how µTorrent starts off. This option is interpreted as a multiplier of 150 bytes, so please enter it as such. Any value below 1 will be ignored (and 1 will be used instead), and any value above 8 will be ignored (and 8 will be used instead). Effectively, that means that the initial packet sizes selectable by the user are the multiples of 150 bytes between (and including) 150 bytes and 1200 bytes.
- **net.utp_packet_size_interval**: This controls how often uTP alters its packet size in response to network conditions, assuming net.utp_dynamic_packet_size is enabled. This value is interpreted in seconds, so please enter it as such.
- **net.utp_receive_target_delay**: This controls the threshold detected connection receive delay that, if surpassed, will cause µTorrent to throttle back on bandwidth usage. The higher this option is set, the more forgiving µTorrent will be toward connection delays, meaning that it will be less likely to throttle back on bandwidth usage. Receive delay is detected by tracking the changes in the deltas between uTP packet timestamps and packet receive times. This option is interpreted in milliseconds, to please enter it as such.

- **net.utp_target_delay**: This option controls the threshold detected connection send delay that, if surpassed, will cause µTorrent to throttle back on bandwidth usage. The higher this option is set, the more forgiving µTorrent will be toward connection delays, meaning that it will be less likely to throttle back on bandwidth usage. Send delay is the receive delay as observed by recipient uTP peers, which is reported back to the client by the recipient peers. This option is interpreted in milliseconds, so please enter it as such.
- **peer.disconnect_inactive**: Enabling this option tells µTorrent to disconnect from a peer that is not transferring with you after peer.disconnect_inactive_interval seconds of inactivity. A peer gets disconnected by this option only if the connection limit has been reached.
- **peer.disconnect_inactive_interval**: This option sets the amount of time µTorrent should wait before breaking an inactive connection. This value is interpreted in seconds, so please enter it as such. Any value below 300 will be ignored (and 300 will be used instead).
- **peer.lazy_bitfield**: Some ISPs block seeding by looking for the complete bitfield and closing the connection. When enabled, µTorrent does not send the complete bitfield, but a sample of it, so as to prevent blocking of seeding.
- **peer.resolve_country**: Enabling this option tells µTorrent to use an Internet database of IP addresses (a DNSBL) to determine a peer's country. Even if the settings directory contains flags.conf and flags.bmp, this option will take precedence, and the internal flag images will be used instead.
- **queue.dont_count_slow_dl**: Enabling this option tells µTorrent to ignore slow downloading torrent jobs as part of the queue. If a torrent job is downloading at less than the value specified by queue.slow_dl_threshold, it will not prevent the next item in the queue from starting.
- **queue.dont_count_slow_ul**: Enabling this option tells µTorrent to ignore slow uploading torrent jobs as part of the queue. If a torrent job is uploading at less than the value specified by queue.slow_ul_threshold, it will not prevent the next item in the queue from starting.
- **queue.prio_no_seeds**: Enabling this option gives torrent jobs without seeds higher priority when seeding than other torrent jobs.
- **queue.slow_dl_threshold**: The rate below which µTorrent should consider a torrent job to be downloading slowly. If µTorrent is downloading at a rate above this value, it is considered to be actively downloading. This value is interpreted in bytes per second, so please enter it as such.
- **queue.slow_ul_threshold**: The rate below which µTorrent should consider a torrent job to be uploading slowly. If µTorrent is uploading at a rate above this value, it is considered to be actively uploading. This value is interpreted in bytes per second, so please enter it as such.
- **queue.use_seed_peer_ratio**: When this option is enabled, µTorrent will determine the seeding queue order based on the ratio of the number of seeds to the number of peers connected in the swarm. The lower the seed:peer ratio is for a torrent job, the higher priority it will be given in the seeding queue. If a torrent job has 0 peers and queue.dont_count_slow_ul is disabled, it will be given the lowest priority. Otherwise, if the aforementioned option is enabled, the torrent job is treated as if there is 1 peer in the swarm.
- **remove_torrent_files_with_private_data**: This option deletes torrents in the .torrent files directory that are not loaded into client.
- **remove_torrent_files_with_private_data**: Unused
- **resume.dir_only**: Unused
- **resume.enable_resume_dir**: Unused
- **rss.feed_as_default_label**: When this option is enabled, µTorrent will use an RSS feed's name as the default label for any torrent jobs added without a label from the RSS feed.
- **rss.smart_repack_filter**: This option tells µTorrent to select an RSS item designated as REPACK over an item without the REPACK designation if both show up in the RSS feed.
- **rss.update_interval**: This option sets the length of time µTorrent should wait between each RSS feed update check. This value is interpreted in minutes, so please enter it as such. Any value below 5 will be ignored (and 5 will be used instead).
- **silent_auto_updates**: Unused
- **sponsored_torrent_offer_enabled**: This options enables new "Offer"-torrents by Bittorrent Inc located at the top of your main view. Only new offers will be disable, and you can skip/cancel them on screen at will. Disable it if you like not to be offered new torrents.
- **store_torr_infohash**: This option saves the .torrent file as INFOHASH.torrent (e.g. ABCDEF1234567890ABCD.torrent).
- **streaming.failover_rate_factor**: Unused
- **streaming.failover_set_percentage**: This option sets the maximum size of the failover set, expressed as percentage of the total number of peers.
- **streaming.min_buffer_piece**: This option sets minimum number of pieces to hold in the streaming buffer.
- **streaming.safety_factor**: Thie option ensures download rate is faster in % than the calculated rate needed.
- **sys.enable_wine_hacks**: This option enables several workarounds for bugs found in Wine (like list-view flickering, or improper display of files list-view in Add New Torrent dialog). This option has no effect on Windows. For changes to this option to take effect, you must restart µTorrent. This applies only if we're on Wine.

- **webui.allow_pairing**: This option allows device/service to pair with the client and control it via WebUI.
- **webui.ssdp_uuid**: TBD
- **webui.token_auth**: This option enables the token authentication system for the Web UI, which is a method for preventing cross-site request forgery attacks that use the authenticated browser session to issue commands to µTorrent. This option breaks backwards compatibility with applications that are unaware of the token system.
- **webui.token_auth_filter**: If token_auth = true, then this **token_auth_filter** changes the scope of token authentication: 0 means that it applies to all connections, 1 means that it applies only to remote connections (i.e. not localhost).

Previous Next